第10章 处 理 器

Linux 运行在若干处理器上，本章简要描述它们。

10.1 X86

TBD

10.2 ARM

ARM处理器实现了一种低功耗、高性能的 32位RISC体系结构，它被广泛使用于嵌入式设备如手机和 PDA(个人式数字化助手)中，它具有 31个32位寄存器，其中 16个在任何模式中均为可见的。它的指令是简单的装载存储指令 (从存储器中装载一个值、执行一个操作、把结果存储进存储器)。它的一个有趣特征是每个指令都是条件式的，例如：你可以测试一个寄存器的值，在此过程中，可以执行你想执行的任何指令，直到在同样条件下测试下一个值为止。另一个有趣的特征是当你装载值时，可以对值执行算术和移位操作，这可以在几种模式下进行，包括系统模式(可从用户模式通过SWI，即软中断进入到系统模式)。

它是一个可合成的核， ARM公司本身不生产处理器，相反 ARM的合伙人(例如Intel公司或LSI公司)在硅片上实现 ARM体系结构，通过一个公共处理器接口，它允许其他处理器紧紧耦合在一起，它有几个存储管理单元的变种，范围从简单的存储器保护格式到复杂的分页层次。

10.3 Alpha AXP处理器

Alpha AXP体系结构是以速度为考虑因素而设计的 64位存取RISC体系结构，所有的寄存器都是64位的，包括32个整数寄存器和 32个浮点寄存器。整数寄存器 31和浮点寄存器31都表示空操作，从中将读出一个 0值，向它们写则不会产生任何影响。所有的指令都是 32位定长，内存操作只有读或写，这种体系结构允许不同的实现方法，只要这种实现方法符合这种体系结构。

指令不能对内存中的数据直接进行操作，所有的数据操作都是在寄存器中进行的。因此，若想增加一个在内存中的计数值，首先把此计数值读入寄存器，然后在寄存器中对其进行修改，最后再写回内存，通过对寄存器或内存的读写指令，各指令间进行数据交互。 Alpha AXP的一个有趣的特征是这些指令能产生标志，例如检验两个寄存器的值是否相等，结果不存入处理器的静态寄存器，而放在第 3个寄存器中。一开始这看起来有些奇怪，但这样做消除了对静态寄存器的依赖性，意味着可以更加容易地建立一个在每一个周期内执行多条指令的CPU，彼此没有联系的寄存器指令并不需要互相等待执行，这与只有一个静态寄存器是不同的。不允许对内存的直接操作和数量庞大的寄存器对建立多指令的执行也是很有帮

助的。

Alpha AXP体系结构使用一组称为特权体系结构库的代码 (Privileged Architecture Library code，PALcode)。PALcode对操作系统、对 Alpha AXP体系结构的CPU实现方法和系统硬件，都是特定的，这些子例程提供了上下文切换、中断、异常和内存管理的操作系统原语，这些子例程能被硬件或被CALL_PAL指令调用。PALcode用标准的Alpha AXP汇编程序编写，包括了对一些实现方法的特殊扩充来提供对下层的硬件函数的直接访问，例如内部处理器寄存器。PALcode运行于PAL模式：一种能制止一些系统事件发生和允许 PALcode完全控制物理系统硬件的特权模式。